arXiv:1210.7756v1 [cs.CR] 29 Oct 2012

A coding theory foundation for the analysis of general unconditionally secure proof-of-retrievability schemes for cloud storage

Maura B. Paterson
Department of Economics, Mathematics and Statistics
Birkbeck, University of London, Malet Street, London WC1E 7HX, UK

Douglas R. Stinson*
David R. Cheriton School of Computer Science
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

Jalaj Upadhyay
David R. Cheriton School of Computer Science
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

October 30, 2012

Abstract

There has been considerable recent interest in "cloud storage" wherein a user asks a server to store a large file. One issue is whether the user can verify that the server is actually storing the file, and typically a challenge-response protocol is employed to convince the user that the file is indeed being stored correctly. The security of these schemes is phrased in terms of an extractor which will recover or retrieve the file given any "proving algorithm" that has a sufficiently high success probability.

This paper treats proof-of-retrievability schemes in the model of unconditional security, where an adversary has unlimited computational power. In this case retrievability of the file can be modelled as error-correction in a certain code. We provide a general analytical framework for such schemes that yields exact (non-asymptotic) reductions that precisely quantify conditions for extraction to succeed as a function of the success probability of a proving algorithm, and we apply this analysis to several archetypal schemes. In addition, we provide a new methodology for the analysis of keyed POR schemes in an unconditionally secure setting, and use it to prove the security of a modified version of a scheme due to Shacham and Waters under a slightly restricted attack model, thus providing the first example of a keyed POR scheme with unconditional security. We also show how classical statistical techniques can be used to evaluate whether the responses of the prover are accurate enough to permit successful extraction. Finally, we prove a new lower bound on storage and communication complexity of POR schemes.

*D. Stinson's research is supported by NSERC discovery grant 203114-11

1 Introduction to Proof-of-retrievability Schemes

In a *proof-of-retrievability scheme* (POR scheme) [3, 6, 8, 12], a user asks a server to store a (possibly large) file. The file is divided into *message blocks* that we view as elements of a finite field. Typically, the file will be "encoded" using a code such as a Reed-Solomon code. The code provides redundancy, enabling erasures or corrupted message blocks to be corrected.

In order for the user to be ensured that the file is being stored correctly on the server, a challenge-response protocol is periodically invoked by the user, wherein the server (the Prover) must give a correct response to a random challenge chosen by the user (the Verifier). This response will typically be a function of one or more message blocks. We do not assume that the user is storing the file. Therefore, in the basic version of a POR scheme, the user must precompute and store a sufficient number of challenge-response pairs, before transmitting the file to the server. After this is done, the user erases the file but retains the precomputed challenge-response pairs. Such schemes are termed *bounded-use* schemes in [6]. An alternative is to use a *keyed* (or *unbounded-use*) scheme, which permits an arbitrary number of challenges to be verified (i.e., the number of challenges does not need to be pre-determined by the user). We will discuss these a bit later. Finally, the user could retain a copy of the file if desired, in which case the responses do not need to be precomputed. This might be done if the server is just being used to store a "back-up" copy of the file.

We wish to quantify the security afforded the user by engaging in the challenge-response protocol (here, "security" means that the user's file can be correctly retrieved by the user). The goal is that a server who can respond correctly to a large proportion of challenges somehow "knows" (or can compute) the contents of the file (i.e., all the message blocks). This is formalised through the notion of an *extractor*, which takes as input a description of a "proving algorithm" $\mathcal{P}$ for a certain unspecified file, and then outputs the file. Actually, for the schemes we study in this paper, the extractor only needs *black-box* access to the proving algorithm. The proving algorithm is created by the server, who is regarded as the adversary in this game. It should be the case that a proving algorithm that is correct with a probability that is sufficiently close to 1 will allow the extractor to determine the correct file. The probability that the proving algorithm $\mathcal{P}$ gives a correct response for a randomly chosen challenge is denoted by $\mathsf{succ}(\mathcal{P})$. We assume that $\mathcal{P}$ always gives some response, so it follows that $\mathcal{P}$ will give an incorrect response with probability $1 - \mathsf{succ}(\mathcal{P})$.

To summarise, we list the components in a POR scheme and the extractor-based security definition we use. Note that we are employing standard models developed in the literature; for a more detailed discussion, see [3, 8, 12].

- The Verifier has a message $m \in (\mathbb{F}_q)^k$ which he redundantly encodes as $M \in (\mathbb{F}_q)^n$.
- M is given to the Prover. In the case of a keyed scheme, the Prover may also be supplied with an additional tag, S.
- The Verifier retains appropriate information to allow him to verify responses. This may or may not include a key K.
- Some number of challenges and responses are carried out by the Prover and Verifier. In each round, the Verifier chooses a challenge c and gives it to the Prover, and the Prover computes a response r which is returned to the Verifier. The Verifier then verifies if the response is correct.
- The computations of the Prover are described in a proving algorithm $\mathcal{P}$.

- The success probability of $\mathcal{P}$ is the probability that it gives a correct response when the challenge is chosen randomly.
- The Extractor is given $\mathcal{P}$ and (in the case of a keyed scheme) K, and outputs an unencoded message $\hat{m}$. Extraction succeeds if $\hat{m} = m$.
- The security of the POR scheme is quantified by proving a statement of the form "the Extractor succeeds with probability at least δ whenever the success probability of $\mathcal{P}$ is at least ϵ". In this paper, we only consider schemes where $\delta = 1$, that is, where extraction is always successful.

1.1 Previous Related Work

Blum et al [2] introduced the concept of memory checking. They formalized a model where performing any sequence of store and request operations on a remote server behaves similarly to local storage. Two seminal papers important to the development of POR schemes are [9, 11]. Lillibridge *et al.* [9] first considered the problem of creating a backup of a large file by redundantly encoding the using an erasure code and distributing pieces of the file to one or more servers. Naor and Rothblum [11] studied *memory checkers* which use message authentication techniques to verify if a file is stored correctly on a remote server. They also consider *authenticators*, which allow a verifier to interact with the server and reconstruct the file provided that a sufficient number of "audits" are all correct.

The concept of *proof-of-retrievability* is due to Juels and Kaliski [8]. A POR scheme incorporates a challenge-response protocol in which a verifier can check that a file is being stored correctly, along with an *extractor* that will actually reconstruct the file, given the algorithm of a "prover" who is able to correctly respond to a sufficiently high percentage of challenges.

There are also papers that describe the closely related (but slightly weaker) idea of a *proof-of-data-possession scheme* (PDP scheme), e.g., [1]. A PDP scheme permits the possibility that not all of the message blocks can be reconstructed. Atieniese *et al.* [1] also introduced the idea of using *homomorphic authenticators* to reduce the communication complexity of the system. Shacham and Waters [12] showed that the scheme of Ateniese *et al.* can be transformed in to a POR scheme by constructing an extractor that extracts the file from the responses of the server on audits.

Bowers, Juels, and Oprea [3] used error-correcting codes, in particular the idea of an "outer" and an "inner" code (in much the same vein as concatenated codes), to get a good balance between the server storage overhead and computational overhead in responding to the audits. A paper that discusses a coding-theoretic approach in the setting of storage enforcement is [7]; this paper makes use of list-decoding techniques, but it concentrates on the storage requirements of the server.

Dodis, Vadhan and Wichs [6] provide the first example of an unconditionally secure POR scheme, also constructed from an error-correcting code, with extraction performed through list decoding in conjunction with the use of an almost-universal hash function. We discuss this particular paper further in Section 1.3.

1.2 Our Contributions

In this paper, we treat the general construction of extractors for POR schemes, with the aim of establishing the precise conditions under which extraction is possible in the setting of *unconditional security*, where the adversary is assumed to have unlimited computational capabilities. In this setting, it turns out that extraction can be interpreted naturally as nearest-neighbour decoding in

a certain code (which we term a "response code"). Previously, error-correcting codes have been used in specific constructions of POR schemes; here, we propose that error-correcting codes constitute the natural foundation to construct as well as analyse arbitrary POR schemes.

There are several advantages of studying unconditionally secure POR schemes. First, the schemes are easier to understand and analyse because we are not making use of any additional cryptographic primitives or unproven assumptions (e.g., PRFs, signatures, bilinear pairings, MACS, hitting samplers, random oracle model, etc.). This allows us to give very simple exact analyses of various schemes. Secondly, the essential role of error-correcting codes in the design and analysis of POR schemes becomes clear: codes are not just a method of constructing POR schemes; rather, every POR scheme gives rise to a code in a natural way.

The success of the extraction process usually depends on the distance of the code used to initially encode the file; when the distance of this code is increased, the extraction process will be successful for less successful provers, thus increasing the security afforded the user. As we mentioned earlier, we quantify the security of a POR scheme by specifying a value ϵ and proving that the extraction process will *always* be able to extract the file, given a prover $\mathcal{P}$ with success probability $\mathsf{succ}(\mathcal{P}) > \epsilon$. (In some other papers, weaker types of extractors are studied, such as extractors that succeed with some specified probability less than 1, or extractors that only recover some specified fraction of the original data.) This allows us to derive conditions which guarantee that extraction will succeed, and to compute exact (or tight) bounds, as opposed to the mainly asymptotic bounds appearing in [6].

We exemplify our approach by considering several archetypal POR schemes. We consider keyed schemes as well as keyless schemes. In Section 2, we first consider the fundamental case where the server is just required to return one or more requested message blocks. Then we progress to a more general scheme where the server must compute a specified linear combination of certain message blocks. Both of these are keyless schemes.

In Section 3, we investigate the Shacham-Waters scheme [12], which is a keyed scheme, modified appropriately to fit the setting of unconditional security. For this scheme, we note that unconditional security can be achieved only if the prover does not have access to a verification oracle. It is also necessary to analyse the success probability of a proving algorithm in the average case, over the set of keys that are consistent with the information given to the prover. This new analytical approach is the first to allow the construction of a secure keyed POR scheme in the unconditionally secure setting.

In Section 4, we look more closely at the numerical conditions we have derived for the various schemes we studied and we provide some useful comparisons and estimates.

We desire that successful extraction can be accomplished whenever $\mathsf{succ}(\mathcal{P})$ exceeds some prespecified threshold. But this raises the question as to how the user is able to determine (or estimate) $\mathsf{succ}(\mathcal{P})$. In many practical POR schemes, the only interaction a user has with the server is through the challenge-response protocol. We show in Section 5 that classical statistical techniques can be used to provide a systematic basis for evaluating whether the responses of the prover are accurate enough to permit successful extraction.

The main overhead of keyed (unbounded-use) unconditionally secure schemes (relative to computationally secure schemes) is in the storage requirements of the user. While this may be problematic in some circumstances, there are significant applications, such as remote backup services, for which such schemes do genuinely represent a practical solution. We also show in Section 6 that a significant additional storage requirement cannot be avoided in this setting, by proving a new

information-theoretic lower bound on storage and communication requirements of POR schemes. Our new bound is an improvement of the information-theoretic lower bound for memory checkers and authenticators proven in [11].

We note that our goal is not to identify a single "best" POR scheme, but rather to provide a useful new methodology to analyse the exact security of various POR schemes in the unconditionally secure setting. The POR problem is one that lends itself naturally to analysis in the unconditionally secure model; the schemes we consider are natural examples of POR protocols that do not require cryptographic building blocks depending on computational assumptions.

For reference, we provide a list of notation used in this paper in Table 3 in the Appendix.

1.3 Comparison with Dodis, Vadhan and Wichs [6]

Our work is most closely related to that of Dodis, Vadhan and Wichs [6]. In [6], it is stated that "there is a clear relation between our problem and the erasure/error decoding of error-correcting codes". Our paper is in some sense a general exploration of these relations, whereas [6] is mainly devoted to a specific construction for a POR scheme that is based on Reed-Solomon codes.

We can highlight several differences between our approach and that of [6]:

- In the setting of unconditional security, the paper [6] only provides bounded-use schemes. Our scheme is the first unbounded-use scheme in this setting.
- The paper [6] mainly uses a (Reed-Solomon) code to construct a specific POR scheme. In contrast, we are studying the connections between an arbitrary POR scheme and the distance of the (related) code that describes the behaviour of the scheme on the possible queries that can be made to the scheme. Stated another way, our approach is to derive a code from a POR scheme, and then to prove security properties of the POR scheme as a consequence of properties of this code.
- The paper [6] uses various tools and algorithms to construct their POR schemes, including Reed-Solomon codes, list decoding, almost-universal hash families, and hitting samplers based on expander graphs. We just use an error-correcting code in our analyses.
- We base our analyses on nearest-neighbour decoding (rather than list decoding, which was used in [6]), and we present conditions under which extraction will succeed with probability equal to 1 (in [6], extraction succeeds with probability close to 1, depending in part on properties of a certain class of hash functions used in the protocol).
- The "POR codes" in [6] are actually protocols that consist of challenges and responses involving a prespecified number of message blocks; we allow challenges in which the responses depend on an arbitrary number of message blocks.
- All our analyses are exact and concrete, whereas the analyses in [6] are asymptotic.

2 Analysis of Several Keyless Schemes

2.1 A Basic Scheme

As a "warm-up", we illustrate our coding theory approach by analysing a simple POR scheme, which we call the Basic Scheme. From now on, we usually refer to the user as the Verifier and the

Figure 1: Basic Scheme

initialisation

Given a *message* $m \in \mathcal{M}$, encode M as $e(m) = M \in \mathcal{M}^*$. The *message encoding function* $e : \mathcal{M} \to \mathcal{M}^*$ is a public bijection. We will assume that $\mathcal{M} = (\mathbb{F}_q)^k$ and $\mathcal{M}^* \subseteq (\mathbb{F}_q)^n$, where q is a prime power and $n \geq k$. We write $M = (m_1, \ldots, m_n)$, where the components $m_1, \ldots, m_n$ are termed *message blocks*.

The Verifier gives the *encoded message* M to the Prover. The Verifier also generates a random *challenge* $c \in \{1, \ldots, n\}$ and stores c and the message block m_c.

challenge-response

The Verifier gives the challenge c to the Prover. The Prover responds with the message block $r = m_c$. The Verifier checks that the *response* r returned by the Prover matches the stored value m_c.

server as the Prover. The basic scheme is presented in Figure 1.

Here is the adversarial model we use to analyse the security of a keyless POR scheme:

1. The Adversary is given the set of encoded messages $\mathcal{M}^*$.
2. The Adversary selects $m \in \mathcal{M}$.
3. The Adversary outputs a deterministic[1] *proving algorithm* for m, denoted $\mathcal{P}$.

The *success probability* of $\mathcal{P}$ is defined to be

$$\mathsf{succ}(\mathcal{P}) = \mathbf{Pr}[\mathcal{P}(c) = m_c],$$

where $e(m) = (m_1, \ldots, m_n)$ and this probability is computed over a challenge $c \in \{1, \ldots, n\}$ chosen uniformly at random.

Now we wish to construct an Extractor that will take as input a proving algorithm $\mathcal{P}$ for some unknown message m. The Extractor will output a message $\widehat{m} \in \mathcal{M}$. We say that the POR scheme is *secure* provided that $\widehat{m} = m$ whenever the success probability of $\mathcal{P}$ is sufficiently close to 1.

The Extractor for the Basic Scheme is presented in Figure 2.

Theorem 2.1. *Suppose that $\mathcal{P}$ is a proving algorithm for the Basic Scheme for which $\mathsf{succ}(\mathcal{P}) > 1 - d/(2n)$, where the hamming distance of the set of encoded messages $\mathcal{M}^*$ is d. Then the Extractor presented in Figure 2 will always output $\widehat{m} = m$.*

Proof. Let M' be the n-tuple of responses computed by $\mathcal{P}$ and denote $\delta = \mathsf{dist}(M, M')$, where $M = e(m)$. Denote $\epsilon = \mathsf{succ}(\mathcal{P})$. Then it is easy to see that $\epsilon = 1 - \delta/n$. We want to prove that $\widehat{M} = M$. We have that $\widehat{M}$ is a vector in $\mathcal{M}^*$ closest to M'. Since M is a vector in $\mathcal{M}^*$ such that $\mathsf{dist}(M, M') = \delta$, it must be the case that $\mathsf{dist}(\widehat{M}, M') \leq \delta$. By the triangle inequality, we get

$$\mathsf{dist}(M, \widehat{M}) \leq \mathsf{dist}(M, M') + \mathsf{dist}(\widehat{M}, M') \leq \delta + \delta = 2\delta.$$

[1] We note that, without loss of generality, the proving algorithm can be assumed to be deterministic. This follows from the observation that any probabilistic proving algorithm can be replaced by a deterministic algorithm relative to which the success of the extractor defined in Figure 2 will not be increased.

Figure 2: Extractor for the Basic Scheme

1. On input $\mathcal{P}$, compute the vector $M' = (m'_1, \ldots, m'_n)$, where $m'_c = \mathcal{P}(c)$ for all $c \in \{1, \ldots, n\}$ (i.e., m'_c is the response computed by $\mathcal{P}$ when it is given the challenge c).
2. Find $\widehat{M} \in \mathcal{M}^*$ so that $\mathsf{dist}(M', \widehat{M})$ is minimised, where $\mathsf{dist}(\cdot, \cdot)$ denotes the hamming distance between two vectors.
3. Output $\widehat{m} = e^{-1}(\widehat{M})$.

However,

$$2\delta = 2n(1-\epsilon) < 2n\left(\frac{d}{2n}\right) = d.$$

Since M and $\widehat{M}$ are vectors in $\mathcal{M}^*$ within distance d, it follows that $M = \widehat{M}$ and the Extractor outputs $m = e^{-1}(M)$, as desired. □

2.2 General Keyless Challenge-Response Schemes

In the simple protocol we analysed, a response to a challenge was just a particular message block chosen from an encoded message M. We are interested in studying more complicated protocols. For example, we might consider a response that consists of several message blocks from M, or is computed as a function of one or more message blocks. In this section, we generalise the preceding extraction process to handle arbitrary keyless challenge-response protocols.

In general, a challenge will be chosen from a specified *challenge space* Γ, and the response will be an element of a *response space* Δ. The *response function* $\rho : \mathcal{M}^* \times \Gamma \to \Delta$ computes the response $r = \rho(M, c)$ given the encoded message M and the challenge c.

For an encoded message $M \in \mathcal{M}^*$, we define the *response vector*

$$r^M = (\rho(M, c) : c \in \Gamma).$$

That is, r^M contains all the responses to all possible challenges for the encoded message M. Finally, define the *response code* (or more simply, the *code*) of the scheme to be

$$\mathcal{R}^* = \{r^M : M \in \mathcal{M}^*\}.$$

The codewords in $\mathcal{R}^*$ are just the response vectors that we defined above.

The Generalised Scheme is presented in Figure 3. Observe that $\mathcal{R}^* \subseteq \Delta^\gamma$, where $\gamma = |\Gamma|$. We will assume that the mapping $M \mapsto r^M$ is an injection, and therefore the hamming distance of $\mathcal{R}^*$ is greater than 0 (in fact, we make this assumption for all the schemes we consider in this paper).

We now describe the generalised adversarial model.

1. The Adversary is given the response code $\mathcal{R}^*$.
2. The Adversary selects $m \in \mathcal{M}$.

Figure 3: Generalised Scheme

initialisation

Given a *message* $m \in \mathcal{M}$, encode m as $e(m) = M \in \mathcal{M}^*$. The Verifier gives M to the Prover. The Verifier also generates a random challenge $c \in \Gamma$ and stores c and $\rho(M, c)$.

challenge-response

The Verifier gives the challenge c to the Prover. The Prover responds with $r = \rho(M, c)$. The Verifier checks that the value r returned by the Prover matches the stored value $\rho(M, c)$.

Figure 4: Extractor for the Generalised Scheme

1. On input $\mathcal{P}$, compute the vector $R' = (r'_c : c \in \Gamma)$, where $r'_c = \mathcal{P}(c)$ for all $c \in \Gamma$ (i.e., for every c, r'_c is the response computed by $\mathcal{P}$ when it is given the challenge c).
2. Find $\widehat{M} \in \mathcal{M}^*$ so that $\mathsf{dist}(R', r^{\widehat{M}})$ is minimised.
3. Output $\widehat{m} = e^{-1}(\widehat{M})$.

3. The Adversary outputs a deterministic proving algorithm for m, denoted $\mathcal{P}$.

The *success probability* of $\mathcal{P}$ is defined to be

$$\mathsf{succ}(\mathcal{P}) = \mathbf{Pr}[\mathcal{P}(c) = \rho(M, c)],$$

where $M = e(m)$ and this probability is computed over a challenge $c \in \Gamma$ chosen uniformly at random.

Now we construct an Extractor that will take as input a proving algorithm $\mathcal{P}$ for some unknown message m. The Extractor will output a message $\widehat{m} \in \mathcal{M}$. We say that the POR scheme is *secure* provided that $\widehat{m} = m$ whenever the success probability of $\mathcal{P}$ is sufficiently close to 1. The Extractor for the Generalised Scheme is presented in Figure 4.

The following theorem relates the success probability of the extractor to the hamming distance of the response code. Its proof is essentially identical to the proof of Theorem 2.1.

Theorem 2.2. *Suppose that $\mathcal{P}$ is a proving algorithm for the Generalised Scheme for which $\mathsf{succ}(\mathcal{P}) > 1 - d^*/(2\gamma)$, where the hamming distance of the response code $\mathcal{R}^*$ is $d^* > 0$. Then the Extractor presented in Figure 4 will always output $\widehat{m} = m$.*

Proof. Let R' be the γ-tuple of responses computed by $\mathcal{P}$ and denote $\delta = \mathsf{dist}(r^M, R')$, where $M = e(m)$. Denote $\epsilon = \mathsf{succ}(\mathcal{P})$. Then it is easy to see that $\epsilon = 1 - \delta/\gamma$. We want to prove that $\widehat{M} = M$. We have that $r^{\widehat{M}}$ is a codeword in $\mathcal{R}^*$ closest to R'. Since M is a codeword such that

Figure 5: Multiblock Challenge Scheme

challenge

A challenge is a subset of ℓ indices $J \subseteq \{1, \ldots, n\}$. Therefore, $\Gamma = \{J \subseteq \{1, \ldots, n\}, |J| = \ell\}$ and $\gamma = \binom{n}{\ell}$.

response

Given the challenge $J = \{i_1, \ldots, i_\ell\}$ where $1 \leq i_1 < \cdots < i_\ell \leq n$, the correct response is the ℓ-tuple

$$\rho(M, J) = (m_{i_1}, \ldots, m_{i_\ell}).$$

Suppose the Verifier receives a response $(r_1, \ldots, r_\ell)$. He then checks that $r_j = m_{i_j}$ for $1 \leq j \leq \ell$.

In this scheme, we have $\Delta = (\mathbb{F}_q)^\ell$.

$\mathsf{dist}(r^M, R') = \delta$, it must be the case that $\mathsf{dist}(r^{\widehat{M}}, R') \leq \delta$. By the triangle inequality, we get

$$\mathsf{dist}(r^M, r^{\widehat{M}}) \leq \mathsf{dist}(r^M, R') + \mathsf{dist}(r^{\widehat{M}}, R') \leq \delta + \delta = 2\delta.$$

However,

$$2\delta = 2\gamma(1 - \epsilon) < 2\gamma \left(\frac{d^*}{2\gamma} \right) = d^*.$$

Since r^M and $r^{\widehat{M}}$ are codewords within distance d^*, it follows that $M = \widehat{M}$ and the Extractor outputs $m = e^{-1}(M)$, as desired. □

Observe that the efficacy of this extraction process depends on the *relative distance* of the response code $\mathcal{R}^*$, which equals d^*/γ. In the next subsections, we look at some specific examples of challenge-response protocols.

2.3 Multiblock Challenge Scheme

We present a POR scheme that we term the Multiblock Challenge Scheme in Figure 5 (this is the same as the "Basic PoR Code Construction" from [6]).

Lemma 2.3. *Suppose that the hamming distance of $\mathcal{M}^*$ is d. Then the hamming distance of the response code of the Multiblock Challenge Scheme is $d^* = \binom{n}{\ell} - \binom{n-d}{\ell}$.*

Proof. Suppose that $M, M' \in \mathcal{M}^*$ and $M \neq M'$. Denote $\mathsf{dist}(M, M') = \delta$, $M = (m_1, \ldots, m_n)$ and $M' = (m'_1, \ldots, m'_n)$. It is easy to see that $r^M_J = r^{M'}_J$ if and only if $J \subseteq \{i : m_i = m'_i\}$. From this, it is immediate that $\mathsf{dist}(r^M, r^{M'}) = \binom{n}{\ell} - \binom{n-\delta}{\ell}$. The desired result follows because $\delta \geq d$. □

Theorem 2.4. *Suppose that $\mathcal{P}$ is a proving algorithm for the Multiblock Challenge Scheme for which*

$$\mathsf{succ}(\mathcal{P}) > \frac{1}{2} + \frac{\binom{n-d}{\ell}}{2\binom{n}{\ell}},$$

Figure 6: Linear Combination Scheme

challenge

A challenge is an n-tuple $V = (v_1, \ldots, v_n) \in (\mathbb{F}_q)^n$.

response

Given the challenge $V = (v_1, \ldots, v_n)$, the correct response is

$$\rho(M, V) = V \cdot M = \sum_{i=1}^{n} v_i m_i,$$

where $M = (m_1, \ldots, m_n)$ and the computation is performed in $\mathbb{F}_q$. Suppose the Verifier receives a response $r \in \mathbb{F}_q$. He then checks that $r = V \cdot M$.

In this scheme, $\Delta = \mathbb{F}_q$.

where the hamming distance of $\mathcal{M}^*$ is d. Then the Extractor presented in Figure 4 will always output $\widehat{m} = m$.

Proof. This is an immediate consequence of Theorem 2.2 and Lemma 2.3, once we verify that

$$1 - \frac{d^*}{2\gamma} = 1 - \frac{\binom{n}{\ell} - \binom{n-d}{\ell}}{2\binom{n}{\ell}} = \frac{1}{2} + \frac{\binom{n-d}{\ell}}{2\binom{n}{\ell}}.$$

□

Remark: When we set $\ell = 1$ in Corollary 2.4, we obtain Theorem 2.1.

2.4 Linear Combination Scheme

In this subsection, we consider the Linear Combination Scheme, in which a response consists of a specified linear combination of message blocks (this could perhaps be thought of as a keyless analogue of the Shacham-Waters scheme [12]). The scheme is presented in Figure 6. We will study two versions of the scheme:

Version 1

Here, the challenge V is any non-zero vector in $(\mathbb{F}_q)^n$, so $\gamma = q^n - 1$.

Version 2

In this version of the scheme, the challenge V is a vector in $(\mathbb{F}_q)^n$ having hamming weight equal to ℓ, so $\gamma = \binom{n}{\ell}(q-1)^\ell$.

2.4.1 Analysis of Version 1

Lemma 2.5. *Suppose that the hamming distance of $\mathcal{M}^*$ is d. The hamming distance of the response code of version 1 of the Linear Combination Scheme is $d^* = q^n - q^{n-1} - 1$.*

Proof. Suppose $M, M' \in \mathcal{M}^*$ and $M \neq M'$. Then $V \cdot M = V \cdot M'$ if and only if $V \cdot (M - M') = 0$. Since $M \neq M'$, there are q^{n-1} solutions for V, given M and M'. There are $q^n - 1$ choices for V, so the desired result follows. □

We observe that d^* is independent of d (the hamming distance of $\mathcal{M}^*$) in version 1 of the scheme.

Theorem 2.6. *Suppose that $\mathcal{P}$ is a proving algorithm for version 1 of the Linear Combination Scheme for which*

$$\mathsf{succ}(\mathcal{P}) > \frac{1}{2} + \frac{1}{2}\left(\frac{q^{n-1}}{q^n - 1}\right).$$

Then the Extractor presented in Figure 4 will always output $\widehat{m} = m$.

2.4.2 Analysis of Version 2

Lemma 2.7. *Suppose that $r \geq 1$ and $X \in (\mathbb{F}_q)^r$ has hamming weight equal to r. Then the number of solutions $V \in (\mathbb{F}_q)^r$ to the equation $V \cdot X = 0$ in which V has hamming weight equal to r, which we denote by a_r, is given by the formula*

$$a_r = \frac{q-1}{q}((q-1)^{r-1} - (-1)^{r-1}). \tag{1}$$

Proof. We prove the result by induction on r. When $r = 1$, there are no solutions, so $a_1 = 0$, agreeing with (1). Now assume that (1) gives the number of solutions for $r = s - 1$, and consider $r = s$. Let $X = (x_1, \ldots, x_s)$ and define $X' = (x_1, \ldots, x_{s-1})$. By induction, the number of solutions to the equation $V' \cdot X' = 0$ in which V' has hamming weight $s - 1$ is a_{s-1}. Each of these solutions V' can be extended to a solution of the equation $V \cdot X = 0$ by setting $v_s = 0$; in each case, the resulting V has hamming weight equal to $s-1$. However, any other vector V' of weight $s-1$ can be extended to a solution of the equation $V \cdot X = 0$ which has hamming weight equal to s. Therefore, we have

$$\begin{aligned}
a_s &= (q-1)^{s-1} - a_{s-1} \\
&= (q-1)^{s-1} - \left(\frac{q-1}{q}((q-1)^{s-2} - (-1)^{s-2})\right) \\
&= (q-1)^{s-1}\left(1 - \frac{1}{q}\right) + \frac{q-1}{q}(-1)^{s-2} \\
&= \frac{q-1}{q}((q-1)^{s-1} - (-1)^{s-1}).
\end{aligned}$$

□

Remark. An alternative way to prove (1) is to observe that a_r is equal to the number of codewords of weight r in an MDS code having length r, dimension $r - 1$ and distance 2. Then (1) can be derived from well-known formulas for the weight distribution of an MDS code. For example, in [10, Ch. 6, Theorem 6], it is shown that the number of codewords of weight w in an MDS code of length n, dimension k and distance $d = n - k + 1$ over $\mathbb{F}_q$ is

$$a_w = \binom{n}{w}(q-1)\sum_{j=0}^{w-d}(-1)^j\binom{w-1}{j}q^{w-d-1}.$$

If we substitute $n = w = r$, $d = 2$ into this formula, we get

$$\begin{aligned} a_r &= \binom{r}{r}(q-1)\sum_{j=0}^{r-2}(-1)^j\binom{r-1}{j}q^{r-2-j} \\ &= \frac{q-1}{q}\sum_{j=0}^{r-2}\binom{r-1}{j}(-1)^j q^{r-1-j} \\ &= \frac{q-1}{q}\left(\sum_{j=0}^{r-1}\binom{r-1}{j}(-1)^j q^{r-1-j} - (-1)^{r-1}\right) \\ &= \frac{q-1}{q}\left((q-1)^{r-1} - (-1)^{r-1}\right), \end{aligned}$$

which agrees with (1), as proven in Lemma 2.7.

We will now compute the distance d^* of the response code $\mathcal{M}^*$. Here is a lemma that will be of use in computing d^*.

Lemma 2.8. *Suppose that $M, M' \in \mathcal{M}^*$ and $M \neq M'$. Denote $\delta = \mathsf{dist}(M, M')$. Let r^M and $r^{M'}$ be the corresponding vectors in the response code of Version 2 of the Linear Combination Scheme. Then*

$$\mathsf{dist}(r^M, r^{M'}) = (q-1)^\ell\left(\binom{n}{\ell} - \binom{n-\delta}{\ell}\right) - \sum_{w\geq 1}\binom{\delta}{w}\binom{n-\delta}{\ell-w}(q-1)^{\ell-w}a_w, \tag{2}$$

where the a_w's are given by (1).

Proof. Suppose that $M, M' \in \mathcal{M}^*$ and $M \neq M'$. Denote

$$M = (m_1, \ldots, m_n) \text{ and } M' = (m'_1, \ldots, m'_n),$$

and let $\delta = \mathsf{dist}(M, M')$. Let

$$J = \{i : m_i = m'_i\} \text{ and } J' = \{1, \ldots, n\} \setminus J.$$

Observe that $|J| = n - \delta$ and $|J'| = \delta$. For any $V = (v_1, \ldots, v_n)$ having hamming weight equal to ℓ, define

$$J_V = \{j \in J : v_j \neq 0\} \text{ and } J'_V = \{j \in J' : v_j \neq 0\}.$$

Denote $w = |J'_V|$; then $|J_V| = \ell - w$.

Suppose $w \geq 1$. Then, given J_V and J'_V, the number of solutions to the equation $V \cdot M = V \cdot M'$ is precisely $(q-1)^{\ell-w}a_w$. When $w = 0$, the number of solutions is $(q-1)^\ell$. Summing over w, and considering all possible choices for J_V and J'_V, we see that the total number of solutions to the equation $V \cdot M = V \cdot M'$ is

$$\binom{n-\delta}{\ell}(q-1)^\ell + \sum_{w\geq 1}\binom{\delta}{w}\binom{n-\delta}{\ell-w}(q-1)^{\ell-w}a_w. \tag{3}$$

The desired result follows. ∎

We can obtain a very accurate estimate for d^* by observing that

$$a_w \approx \frac{(q-1)^w}{q}$$

is a very accurate approximation. After making this substitution, it is easy to see that the quantity (3) is minimised when $\delta = d$. This minimises (2), so we obtain

$$d^* \approx \binom{n}{\ell}(q-1)^\ell - \binom{n-d}{\ell}(q-1)^\ell - \sum_{w\geq 1}\binom{d}{w}\binom{n-d}{\ell-w}\frac{(q-1)^\ell}{q}. \tag{4}$$

We have

$$\begin{aligned}\sum_{w\geq 1}\binom{d}{w}\binom{n-d}{\ell-w}\frac{(q-1)^\ell}{q} &= \frac{(q-1)^\ell}{q}\sum_{w\geq 1}\binom{d}{w}\binom{n-d}{\ell-w} \\ &= \frac{(q-1)^\ell}{q}\left(\binom{n}{\ell} - \binom{n-d}{\ell}\right).\end{aligned}$$

Therefore, from (4), we get

$$\begin{aligned}d^* &\approx (q-1)^\ell\left(\binom{n}{\ell} - \binom{n-d}{\ell}\right) - \frac{(q-1)^\ell}{q}\left(\binom{n}{\ell} - \binom{n-d}{\ell}\right) \\ &= \frac{(q-1)^{\ell+1}}{q}\left(\binom{n}{\ell} - \binom{n-d}{\ell}\right).\end{aligned} \tag{5}$$

The following theorem uses the estimated value for d^* derived in (5).

Theorem 2.9. *Suppose that $\mathcal{P}$ is a proving algorithm for version 2 of the Linear Combination Scheme for which*

$$\mathsf{succ}(\mathcal{P}) > \frac{1}{2} + \frac{1}{2}\left(\frac{1}{q} + \frac{(q-1)\binom{n-d}{\ell}}{q\binom{n}{\ell}}\right),$$

where the hamming distance of $\mathcal{M}^*$ is d. Then the Extractor presented in Figure 4 will always output $\widehat{m} = m$.

3 Analysis of a Keyed Scheme: the Shacham-Waters Scheme

The Shacham-Waters Scheme [12] is a *keyed proof-of-retrievability scheme*. This means that the Verifier has a secret key that is not provided to the Prover. This key is used to verify responses in the challenge-response protocol, and it is also provided to an extraction algorithm as input. The use of a key permits an arbitrary number of challenges to be verified, without the Verifier having to precompute the responses.

We discuss a variation of the Shacham-Waters Scheme [12], modified to fit the unconditional security setting. The main change is that the vector $(\beta_1, \ldots, \beta_n)$ (which comprises part of the key) is completely random, rather than being generated by a pseudorandom function (i.e., a PRF). This scheme, presented in Figure 7, is termed the Modified Shacham-Waters Scheme.

As we did with the Linear Combination Scheme, we will study two versions of the scheme. In the first version, the challenge V is any non-zero vector in $(\mathbb{F}_q)^n$, so $\gamma = q^n - 1$. In the second version,

Figure 7: Modified Shacham-Waters Scheme

- The *key* K consists of $\alpha \in \mathbb{F}_q$ and $B = (\beta_1, \ldots, \beta_n) \in (\mathbb{F}_q)^n$. K is retained by the Verifier.
- The *encoded message* is $M = (m_1, \ldots, m_n) \in (\mathbb{F}_q)^n$.
- The *tag* is $S = (\sigma_1, \ldots, \sigma_n) \in (\mathbb{F}_q)^n$, where S is computed using the following (vector) equation in $\mathbb{F}_q$:
$$S = B + \alpha M. \tag{6}$$
The message M and the tag S are given to the Prover.
- A *challenge* is a vector $V = (v_1, \ldots, v_n) \in (\mathbb{F}_q)^n$.
- The *response* consists of $(\mu, \tau) \in (\mathbb{F}_q)^2$, where the following computations are performed in $\mathbb{F}_q$:
$$\mu = V \cdot M \tag{7}$$
and
$$\tau = V \cdot S. \tag{8}$$
- The response (μ, τ) is *verified* by checking that the following condition holds in $\mathbb{F}_q$:
$$\tau \stackrel{?}{=} \alpha\mu + V \cdot B. \tag{9}$$

the challenge V is a vector in $(\mathbb{F}_q)^n$ having hamming weight equal to ℓ, so $\gamma = \binom{n}{\ell}(q-1)^\ell$. In both versions, $\Delta = (\mathbb{F}_q)^2$.

The information held by the various parties in the scheme is summarised as follows:

Verifier	Prover	Extractor
$K = (\alpha, B)$	M, S	$K, \mathcal{P}$

We first observe that, from the point of view of the Prover, there are q possible keys.

Lemma 3.1. *Given M and S, the Prover can restrict the set of possible keys (α, B) to*

$$\mathsf{Possible}(M, S) = \{(\alpha_0, S - \alpha_0 M) : \alpha_0 \in \mathbb{F}_q\}.$$

Proof. Suppose that $\alpha = \alpha_0$. Then equation (6) implies that $B = S - \alpha_0 M$. □

We will define a response (μ, τ) to be *acceptable* if (9) is satisfied. A response is *authentic* if it was created using equations (7) and (8). Note that an authentic response will be acceptable for every key $K \in \mathsf{Possible}(M, S)$. In the case of an acceptable (but perhaps not authentic) response, we have the following useful lemma.

Lemma 3.2. *Suppose that a response (μ, τ) to a challenge V for a message M is acceptable for more than one key in* $\mathsf{Possible}(M, S)$*. Then (μ, τ) is authentic.*

Proof. Suppose $K_1, K_2 \in \mathsf{Possible}(M, S)$, where $K_1 = (\alpha_1, B_1)$, $K_2 = (\alpha_2, B_2)$ and $\alpha_1 \neq \alpha_2$. We have $B_1 = S - \alpha_1 M$ and $B_2 = S - \alpha_2 M$. Now consider a response (μ, τ) to a challenge V that is acceptable for both of the keys K_1 and K_2. Then

$$\tau = \alpha_1 \mu + V \cdot B_1 = \alpha_2 \mu + V \cdot B_2.$$

Therefore,

$$(\alpha_1 - \alpha_2)\mu + V \cdot (B_1 - B_2) = 0.$$

However, we have $B_1 - B_2 = (\alpha_2 - \alpha_1)M$, so

$$(\alpha_1 - \alpha_2)(\mu - V \cdot M) = 0.$$

We have $\alpha_1 \neq \alpha_2$, so it follows that $\mu = V \cdot M$. Then we obtain

$$\begin{aligned} \tau &= \alpha_1 \mu + V \cdot B_1 \\ &= \alpha_1 V \cdot M + V \cdot (S - \alpha_1 M) \\ &= V \cdot S. \end{aligned}$$

Therefore the response (μ, τ) is authentic. □

The verification condition (9) depends on the key, but not on the message M. It follows that the Prover can create acceptable non-authentic responses if he knows the value of α (which in turn uniquely determines the key, as shown in Lemma 3.1). This leads to the following attack.

Theorem 3.3. *If the Prover has access to a verification oracle, then the Modified Shacham-Waters Scheme is not unconditionally secure.*

Proof. For every key $K \in \mathsf{Possible}(M, S)$, the Prover can create a response (μ, τ) to a challenge V that will be acceptable if and only if K is the actual key (this follows from Lemma 3.2). The Prover can check the validity of these responses by accessing the verification oracle. As soon as one of these responses is accepted by the verification oracle, the Prover knows the correct value of the key. Hence the Prover can now create a proving algorithm $\mathcal{P}$ that will output acceptable but non-authentic responses. This algorithm $\mathcal{P}$ will not allow the correct message to be extracted.

In more detail, after the Prover has determined the key $K = (\alpha, B)$, he chooses an arbitrary (encoded) message $M' \neq M$ and constructs $\mathcal{P}$ as follows:

1. Given a challenge V, define $\mu = V \cdot M'$.
2. Then define $\tau = \alpha\mu + V \cdot B$.

Suppose the Extractor is run on $\mathcal{P}$. It is easy to see that $\mathsf{dist}(R', r^{M'}) = 0$, so the Extractor will compute $\widehat{M} = M'$, which is incorrect. □

Even in the absence of a verification oracle, the Prover can guess the correct value of α, which he can do successfully with probability $1/q$. If he correctly guesses α, he can create a non-extractable proving algorithm. This implies that it is not possible to prove a theorem stating that *any* proving algorithm yields an extractor. However, we can prove meaningful reductions if we define the success probability of a proving algorithm to be the *average* success probability over the q possible keys that are consistent with the information given to the Prover.

Suppose $\mathcal{P}$ is a (deterministic) proving algorithm for a message $M = (m_1, \ldots, m_n) \in (\mathbb{F}_q)^n$. For each challenge V and for every key $K = (\alpha, B) \in \mathsf{Possible}(M, S)$, define $\chi(V, K) = 1$ if $\mathcal{P}$ returns an acceptable response for the key K given the challenge V, and define $\chi(V, K) = 1$, otherwise.

Since there are γq choices for the pair (V, K), we define the *average success probability* $\mathsf{succ}_{\mathsf{avg}}(\mathcal{P})$ to be

$$\mathsf{succ}_{\mathsf{avg}}(\mathcal{P}) = \frac{\sum_{V \in \Gamma, K \in \mathsf{Possible}(M,S)} \chi(V, K)}{\gamma q}. \tag{10}$$

Lemma 3.4. *Suppose there are D challenges V for which $\mathcal{P}$ returns an authentic response, and hence there are $C = \gamma - D$ challenges for which $\mathcal{P}$ returns a response that is not authentic. Then*

$$\mathsf{succ}_{\mathsf{avg}}(\mathcal{P}) \leq 1 - \frac{C(q-1)}{\gamma q}. \tag{11}$$

Proof. If V is a challenge for which $\mathcal{P}$ returns an authentic response, then $\chi(V, K) = 1$ for every K. If V is a challenge for which $\mathcal{P}$ does not return an authentic response, then Lemma 3.2 implies that $\chi(V, K) = 1$ for at most one K. Therefore,

$$\sum_{V \in \Gamma, K \in \mathsf{Possible}(M,S)} \chi(V, K) \leq C + qD = \gamma q - C(q-1).$$

The desired result now follows from (10). □

Let's now turn our attention to the response code. Even though a response is an ordered pair (μ, τ), it suffices to consider only the values of μ in the extraction process (this follows because μ, K and V uniquely determine τ; see (9)). So we will define the response vector for a message M to be $r^M = (M \cdot V : V \in \Gamma)$. Observe that this response vector is identical to the response vector in the Linear Combination Scheme.

Lemma 3.5. *Suppose that $\mathcal{P}$ is a proving algorithm for the message M in the Modified Shacham-Waters Scheme. Let $r^M = (M \cdot V : V \in \Gamma)$ and let R' be the γ-tuple of responses computed by $\mathcal{P}$. Then*

$$\mathsf{dist}(r^M, R') \leq \frac{(1 - \mathsf{succ}_{\mathsf{avg}}(\mathcal{P}))\gamma q}{q-1}.$$

Proof. Define C as in Lemma 3.4. Since a co-ordinate of r^M differs from the corresponding co-ordinate of R' only when the response is non-authentic, it follows that $\mathsf{dist}(r^M, R') \leq C$. Equation (11) implies that

$$C \leq \frac{(1 - \mathsf{succ}_{\mathsf{avg}}(\mathcal{P}))\gamma q}{q-1},$$

from which the stated result follows. □

Theorem 3.6. *Suppose that*

$$\mathsf{succ}_{\mathsf{avg}}(\mathcal{P}) > 1 - \frac{d^*(q-1)}{2\gamma q}, \tag{12}$$

where d^* is given by equation (2). Then the Extractor presented in Figure 8 will always output $\widehat{m} = m$.

Figure 8: Extractor for the Modified Shacham-Waters Scheme

1. On input $\mathcal{P}$, compute the vector $R' = (\mu_V : V \in \Gamma)$, where $(\mu_V, \tau_V) = \mathcal{P}(V)$ for all $V \in \Gamma$.
2. Find $\widehat{M} \in \mathcal{M}^*$ so that $\mathsf{dist}(R', r^{\widehat{M}})$ is minimised.
3. Output $\widehat{m} = e^{-1}(\widehat{M})$.

Proof. Denote $\epsilon = \mathsf{succ}_{\mathsf{avg}}(\mathcal{P})$, let R' be the γ-tuple of responses computed by $\mathcal{P}$, and denote $\delta = \mathsf{dist}(r^M, R')$, where $M = e(m)$. We showed in Lemma 3.5 that

$$\delta \leq \frac{(1-\epsilon)\gamma q}{q-1}.$$

We want to prove that $\widehat{M} = M$. We have that $r^{\widehat{M}}$ is a codeword in $\mathcal{R}^*$ closest to R'. Since M is a codeword such that $\mathsf{dist}(r^M, R') = \delta$, it must be the case that $\mathsf{dist}(r^{\widehat{M}}, R') \leq \delta$. By the triangle inequality, we get

$$\mathsf{dist}(r^M, r^{\widehat{M}}) \leq \mathsf{dist}(r^M, R') + \mathsf{dist}(r^{\widehat{M}}, R') \leq \delta + \delta = 2\delta.$$

However,

$$2\delta \leq \frac{2(1-\epsilon)\gamma q}{q-1} < d^*,$$

where the last inequality follows from (12). Since r^M and $r^{\widehat{M}}$ are codewords within distance d^* (which is the distance of the response code), it follows that $M = \widehat{M}$ and the Extractor outputs $m = e^{-1}(M)$, as desired. □

4 Numerical Computations and Estimates

We have provided sufficient conditions for extraction to succeed for several POR schemes, based on the success probability of the proving algorithm. Here look a bit more closely at these numerical conditions and provide some useful comparisons and estimates for the different schemes we have studied.

We will consider three schemes. We have the following observations, which can be verified in a straightforward manner:

1. For the Multiblock Challenge Scheme, extraction will succeed if

$$\mathsf{succ}(\mathcal{P}) > S_0 = \frac{1}{2} + \frac{\binom{n-d}{\ell}}{2\binom{n}{\ell}}.$$

This is stated in Theorem 2.4.

2. For the Linear Combination Scheme (Version 2), extraction will succeed if

$$\mathsf{succ}(\mathcal{P}) > S_1 = \left(\frac{q-1}{q}\right) S_0 + \frac{1}{q}.$$

This follows from Theorem 2.9.

3. For the Modified Shacham-Waters Scheme, extraction will succeed if

$$\mathsf{succ}_{\mathsf{avg}}(\mathcal{P}) > S_2 = \left(\frac{q-1}{q}\right)^2 S_0 + \frac{2}{q} - \frac{1}{q^2}.$$

This follows from Theorem 3.6, using the estimate for d^* given in (5).

It is clear that S_0, S_1 and S_2 are extremely close for any reasonable value of q (such as $q \geq 2^{32}$, for example). Therefore we will confine our subsequent analysis to S_0 and state our results in terms of the Multiblock Challenge Scheme. The formula for S_0 is relatively simple, but it is complicated somewhat by the binomial coefficients. Therefore, it may be useful to define an estimate that does not involve binomial coefficients.

Theorem 4.1. *Denote $\epsilon = 1 - \mathsf{succ}(\mathcal{P})$. Suppose that the following inequality holds in the Multiblock Challenge Scheme:*

$$\frac{\ell d}{n} > \ln\left(\frac{1}{1-2\epsilon}\right). \tag{13}$$

Then the Extractor will always succeed.

Proof. From (13), we obtain

$$\ln(1-2\epsilon) > -\frac{\ell d}{n}.$$

Now $-x > \ln(1-x)$ for $0 < x < 1$, we obtain

$$\ln(1-2\epsilon) > \ell \ln\left(1 - \frac{d}{n}\right).$$

Exponentiating both sides of this inequality, we have

$$1 - 2\epsilon > \left(1 - \frac{d}{n}\right)^{\ell}.$$

It is easy to prove that

$$\left(1 - \frac{d}{n}\right)^{\ell} > \frac{\binom{n-d}{\ell}}{\binom{n}{\ell}},$$

so it follows that

$$1 - 2\epsilon > \frac{\binom{n-d}{\ell}}{\binom{n}{\ell}}.$$

From this, we obtain

$$\mathsf{succ}(\mathcal{P}) > \frac{1}{2} + \frac{\binom{n-d}{\ell}}{2\binom{n}{\ell}},$$

and hence the Extractor will always succeed. □

ℓ	d	$\mathsf{succ}(\mathcal{P})$	n (Thm. 2.4)	n (Thm. 4.1)	ℓ	d	$\mathsf{succ}(\mathcal{P})$	n (Thm. 2.4)	n (Thm. 4.1)
10000	10000	0.6	62143493	62133493	10000	1000	0.6	6218850	6213349
		0.7	109145666	109135666			0.7	10919066	10913566
		0.8	195771518	195761518			0.8	19581651	19576151
		0.9	448152011	448142011			0.9	44819701	44814201
		0.99	4949841645	4949831645			0.99	494988664	494983164
1000	10000	0.6	6218850	6213349	1000	1000	0.6	622334	6213349
		0.7	10919066	10913567			0.7	1092356	10913567
		0.8	19581651	19576152			0.8	1958614	19576152
		0.9	44819700	44814201			0.9	4482419	4481420
		0.99	494988664	494983164			0.99	49499315	49498316
100	10000	0.6	626398	621334	100	1000	0.6	62684	62133
		0.7	1096413	1091357			0.7	109685	109135
		0.8	1962669	1957615			0.8	196311	195761
		0.9	4486471	4481420			0.9	448691	448142
		0.99	49503366	49498316			0.99	4950381	4949831
50	10000	0.6	315719	310667	50	1000	0.6	31594	31068
		0.7	550718	5456783			0.7	55093	545678
		0.8	983840	9788076			0.8	98406	978807
		0.9	2245736	2240710			0.9	224599	224071
		0.99	24754183	24749158			0.99	2475440	2474916
10000	100	0.6	626398	621334	10000	10	0.6	67272	62133
		0.7	1096413	1091356			0.7	114216	109136
		0.8	1962668	1957615			0.8	200808	195761
		0.9	4486471	4481420			0.9	453165	448142
		0.99	4950336	49498316			0.99	4954838	4949832
1000	100	0.6	62684	62133	1000	10	0.6	6731	6213
		0.7	109685	109135			0.7	11425	10914
		0.8	196311	195761			0.8	20084	19576
		0.9	448692	448142			0.9	45320	44814
		0.99	4950381	4949831			0.99	495488	494983
100	100	0.6	6313	6213	100	10	0.6	677	621
		0.7	11013	10913			0.7	1146	1091
		0.8	19675	19576			0.8	2012	1958
		0.9	44913	44814			0.9	4536	4481
		0.99	495082	494983			0.99	49552	49498
50	100	0.6	3181	3106	50	10	0.6	341	311
		0.7	5531	5456			0.7	576	546
		0.8	9862	9788			0.8	1009	979
		0.9	22481	22407			0.0	2270	2240
		0.99	247565	247492			0.99	24779	24749

Table 1: Values of n which the Extractor will always succeed.

Table 1 presents values of n (the length of an encoded message), for different values of ℓ (the hamming weight of the challenge), d (the distance of the code) and the success probability of the proving algorithm, such that the Extractor is guaranteed to succeed. We tabulate the value of n as specified by Theorems 2.4 and 4.1. We see, for a wide range of parameters, that the estimate obtained in Theorem 4.1 is very close to the earlier value computed in Theorem 2.4.

5 Estimating the Success Probability of a Prover

5.1 Hypothesis Testing

The essential purpose of a POR scheme is to assure the user that their file is indeed being stored correctly, i.e., in such a manner that the user can recover the entire file if desired. We have considered several schemes for testing whether this is the case, but in order to use these schemes appropriately it is also necessary to pay attention to how we interpret the results of these tests. Theorem 2.1 tells us that extraction is possible for the Basic Scheme whenever $\mathsf{succ}(\mathcal{P})$ is at least $(n - \lfloor \frac{d}{2} \rfloor + 1)/n$, hence the information we would like to obtain from using the Basic Scheme is whether or not $\mathsf{succ}(\mathcal{P})$ exceeds the necessary threshold. Similarly, for the Multiblock Challenge Scheme or the Linear Combination Scheme, we can compute a value ω such that extraction is possible whenever $\mathsf{succ}(\mathcal{P}) > \frac{\omega-1}{\gamma}$. We can calculate $\mathsf{succ}(\mathcal{P})$ for a given proving algorithm $\mathcal{P}$ if we know the values of $\mathcal{P}$'s response $\mathcal{P}(c)$ for every possible challenge $c \in \Gamma$. However, the whole purpose of a POR scheme is to provide reassurance that $\mathsf{succ}(\mathcal{P})$ is sufficiently large without having to request all $\mathcal{P}(c)$ for all $c \in \Gamma$. Given the prover's responses to some subset of possible challenges, the user wishes to make a judgement as to whether he/she is satisfied that $\mathsf{succ}(\mathcal{P})$ is acceptably high. This takes us straight into the realm of classical statistical techniques such as *hypothesis testing* [4, 5].

Suppose the prover has given responses $\mathcal{P}(c)$ to t challenges c chosen uniformly at random without replacement from Γ, and that g of these responses are found to be correct. We are concerned that the prover's success rate may not be high enough, so we are looking for evidence to convince us that in fact it is sufficiently high to permit extraction. In other words, we wish to distinguish the null hypothesis

$H_0 : \mathsf{succ}(\mathcal{P}) \leq \frac{\omega-1}{\gamma}$;

from the alternative hypothesis

$H_1 : \mathsf{succ}(\mathcal{P}) \geq \frac{\omega}{\gamma}$.

Suppose that H_0 is true. Then the probability that the number of correct responses is at least g is itself at most

$$\sum_{i=g}^{t} \frac{\binom{\omega-1}{i}\binom{\gamma-\omega+1}{t-i}}{\binom{\gamma}{t}}. \tag{14}$$

If this probability is less than 0.05, then we reject H_0 and instead accept the alternative hypothesis (namely that $\mathsf{succ}(\mathcal{P})$ is sufficiently high to permit extraction. In this case we conclude that the server is storing the file appropriately.) If the probability is greater than 0.05 then there is insufficient evidence to reject H_0 at the 5% significance level (so we continue to suspect that the server is perhaps not storing the file adequately).

Alternatively, if we choose the challenges uniformly at random *with* replacement, then the condition for rejecting the null hypothesis becomes

$$\sum_{i=g}^{t} \binom{t}{i} \left(\frac{\omega-1}{\gamma}\right)^{i} \left(\frac{\gamma-\omega+1}{\gamma}\right)^{t-i} < 0.05.$$

Example 1. Suppose that for the Basic Scheme $n = 1000$, and that the minimum distance of the response code is 400. Then by Theorem 2.1 we find that extraction is possible whenever $\mathsf{succ}(\mathcal{P})$ is greater than 0.8. Suppose the prover responds to 100 challenges that have been chosen uniformly with replacement, and that 87 of the responses were correct. We find that

$$\sum_{i=87}^{100} \binom{100}{i} 0.8^{i} 0.2^{100-i} \approx 0.047 < 0.05.$$

Thus, in this case there is sufficient evidence to reject the null hypothesis at the 5% significance level, and so we conclude that the file is in fact being stored correctly.

On the other hand, if only 86 of the responses were correct, we observe that

$$\sum_{i=86}^{100} \binom{100}{i} 0.8^{i} 0.2^{100-i} \approx 0.08 > 0.05.$$

In this case there is not enough evidence to reject the null hypothesis at the 5% significance level, and so we continue to suspect that the server is not storing the file adequately.

The benefit of this statistical approach is that given the observed responses to the challenges, for any desired value of α we can construct a hypothesis test for which the probability of inappropriately rejecting the null hypothesis (and hence failing to catch a prover that does not permit extraction) is necessarily less than α.[2] This is the case regardless of the true value of $\mathsf{succ}(\mathcal{P})$, and we do not need to make any *a priori* assumptions about this value.

In Table 2 we give examples of a range of possible results of the challenge process and the corresponding outcomes in terms of whether the null hypothesis is rejected at either the 5% or 1% significance level.

5.2 Confidence Intervals

Another closely-related way to portray the information provided by the sample of responses to challenges is through the use of confidence intervals. We define a 95% lower confidence bound θ_L by

$$\theta_L = \sup \left\{ \theta \Big| \sum_{i=g}^{t} \binom{t}{i} \theta^{i} (1-\theta)^{t-i} < 0.05 \right\}$$

This represents the largest possible value for $\mathsf{succ}(\mathcal{P})$ for which the probability of obtaining g or more correct responses in a sample of size t is less than 0.05. Then the decision process for the hypothesis test described in Section 5.1 consists of rejecting the null hypothesis whenever $\frac{\omega-1}{n} < \theta_L$,

[2] Here we refer to probability over the set of all possible choices of t challenges.

since if $\frac{\omega-1}{n}$ is less than the critical value we know that the probability of a prover with success rate at most $\frac{\omega-1}{n}$ providing g or more correct responses is less than 0.05.

The interval $(\theta_L, 1]$ is a 95% *confidence interval for* $\mathsf{succ}(\mathcal{P})$: if a large number of samples of size t were made and the corresponding intervals were calculated using this approach, then you would expect the resulting intervals to contain the true value of $\mathsf{succ}(\mathcal{P})$ at least 95% of the time. The hypothesis test can be expressed in terms of the confidence interval $(\theta_L, 1]$ by stating that we reject H_0 whenever $\frac{\omega-1}{n}$ does not lie in this interval.

Example 2. Suppose we have $n = 1000$ and $d = 400$ as in Example 1, and suppose that 90 of the responses are correct. Then

$$\begin{aligned}\theta_L &= \sup\left\{\theta \,\middle|\, \sum_{i=90}^{t} \binom{t}{i}\theta^i(1-\theta)^{t-i} < 0.05\right\} \\ &\approx 0.836\end{aligned}$$

Then a 95% confidence interval for $\mathsf{succ}(\mathcal{P})$ is $(0.836, 1]$, and hence we reject the null hypothesis, as $\frac{\omega-1}{n} = 0.8$ does not lie in this interval.

$\frac{\omega-1}{\gamma}$	t	g	$\alpha = 0.05$	$\alpha = 0.01$	$\frac{\omega-1}{\gamma}$	t	g	$\alpha = 0.05$	$\alpha = 0.01$
0.8	100	100	✓	✓	0.9	100	100	✓	✓
0.8	100	95	✓	✓	0.9	100	95	X	X
0.8	100	90	X	X	0.9	100	90	X	X
0.8	100	85	X	X	0.9	100	85	X	X
0.8	100	80	X	X	0.9	100	80	X	X
0.8	200	180	✓	✓	0.9	200	200	✓	✓
0.8	200	175	✓	✓	0.9	200	195	✓	✓
0.8	200	170	✓	X	0.9	200	190	✓	✓
0.8	200	165	X	X	0.9	200	185	X	X
0.8	200	160	X	X	0.9	200	180	X	X
0.8	500	435	✓	✓	0.9	500	480	✓	✓
0.8	500	430	✓	✓	0.9	500	475	✓	✓
0.8	500	425	✓	✓	0.9	500	470	✓	✓
0.8	500	420	✓	X	0.9	500	465	✓	X
0.8	500	415	X	X	0.9	500	460	X	X

Table 2: Outcomes of hypothesis testing for a range of responses. The columns headed by values of α contain a tick if H_0 is rejected at the corresponding significance level, and a cross otherwise.

5.3 Reacting to a Suspect Prover

One question that has not always been directly considered is what action to take when a prover is suspected of cheating. In the framework of Section 5.1 this becomes the problem of what to do in the case where there is insufficient evidence to reject the null hypothesis. There are various possible options at this point, and the choice of option will depend on factors such as the reason for storing the file, and any costs and inconvenience that might be associated with associated with ceasing to

use that server, or with switching to another storage provider. For example, if a server is simply being used as a backup service for non-critical data and there is a high overhead associated with switching storage providers, then a user will not want to be overhasty in taking action against a possibly innocent server. In this case an appropriate action in the first instance might be to seek more responses to challenges in order to avoid the possibility that the earlier set of responses were unrepresentative of the reliability of the prover in general.

5.4 Comparison with Approaches Followed in the Literature

Ateniese et al. [1] observe that if $\mathsf{succ}(\mathcal{P}) \leq \frac{g}{\gamma}$ (with $g \in \mathbb{Z}$) then when $\mathcal{P}$ is queried on t possible challenges chosen uniformly at random without replacement then the probability $\mathsf{Pr}_{\mathsf{bad}}$ that at least one incorrect response is observed is given by

$$\mathsf{Pr}_{\mathsf{bad}} = 1 - \frac{\binom{g}{t}}{\binom{\gamma}{t}},$$

and they note that

$$1 - \left(\frac{g}{\gamma}\right)^t \leq \mathsf{Pr}_{\mathsf{bad}} \leq 1 - \left(\frac{g+1-t}{\gamma-t+1}\right)^t.$$

As examples of parameters, they point out that if $\mathsf{succ}(\mathcal{P}) = 0.99$ then to achieve $\mathsf{Pr}_{\mathsf{bad}} = 0.95$ requires $t = 300$, and $\mathsf{Pr}_{\mathsf{bad}} = 0.99$ requires $t = 460$. They comment that the required number t is in fact independent of γ, sinced it is based instead on the required threshold for $\mathsf{succ}(\mathcal{P})$ -this observation applies equally to our analysis.

Dodis, Vadhan and Wichs [6] use a similar approach to Ateniese et al., but in addition propose the use of a *hitting sampler* that amounts to choosing which t elements to sample from a specified distribution that contains fewer than $\binom{n}{t}$ possible sample sets but still guarantees that $\mathsf{Pr}_{\mathsf{bad}}$ is higher than some specified value for a given value of $\mathsf{succ}(\mathcal{P})$ that is less than 1.

This analysis can be interpreted in the context of a hypothesis test to distinguish the null hypothesis

$H_0 : \mathsf{succ}(\mathcal{P}) \leq 0.99;$

from the alternative hypothesis

$H_1 : \mathsf{succ}(\mathcal{P}) > 0.99.$

If 300 challenges are made and all the responses are correct, then a 95% confidence interval for $\mathsf{succ}(\mathcal{P})$ is $(0.99006, 1]$, so there is enough evidence to reject the null hypothesis at the 5% significance level. However, a 99% confidence interval for $\mathsf{succ}(\mathcal{P})$ is $(0.977, 1]$, so there is insufficient evidence to reject the null hypothesis at the 1% significance level. If, on the other hand, 460 challenges were made and all the responses were correct then a 99% confidence interval for $\mathsf{succ}(\mathcal{P})$ is $(0.99003, 1]$ and so in this case there *is* enough evidence to reject the null hypothesis at the 1% significance level.

We note that this is a special case of the analysis in Sections 5.1 and 5.2. Specifically, Ateniese et al. are focusing on determining the smallest number of challenges for which an entirely correct response constitutes sufficient evidence to reject the null hypothesis at the desired significance level.

While this does result in the smallest number of challenges for which there is still the potential to reassure the user as to the appropriate behaviour of the prover, it has the drawback that even a single incorrect response results in failure to reject the null hypothesis, regardless of whether it is true. Taking a larger sample size has the benefit of increasing the probability that a false null hypothesis is rejected, without adversely affecting the probability that a true H_0 fails to be rejected. For example, from Table 2 we see that if 90 correct responses out of 100 are observed then there is insufficient evidence to reject the hypothesis $\mathsf{succ}(\mathcal{P}) \leq 0.8$ at the 5% significance level. However, if 180 correct responses out of 200 are observed then there *is* sufficient evidence to reject this null hypothesis at the 5% significance level (in fact we even have enough evidence to reject H_0 at the 1% significance level).

6 A Lower Bound on Storage and Communication Requirements

In this section, we prove a bound that applies to *keyed* POR schemes. Suppose that $\mathbf{M}$ is a random variable corresponding to a randomly chosen *unencoded* message m. Let $\mathbf{V}$ be a random variable denoting the information stored by the Verifier (i.e., the key), and let $\mathbf{R}$ be a random variable corresponding to the computations performed by an extractor. It is obvious that the probability that m can correctly be reconstructed is $2^{-H(\mathbf{M}|\mathbf{V},\mathbf{R})}$. Now, from basic entropy inequalities, we have

$$\begin{aligned} H(\mathbf{M}|\mathbf{V},\mathbf{R}) &= H(\mathbf{M},\mathbf{V},\mathbf{R}) - H(\mathbf{V},\mathbf{R}) \\ &\geq H(\mathbf{M},\mathbf{V},\mathbf{R}) - H(\mathbf{V}) - H(\mathbf{R}) \\ &\geq H(\mathbf{M}) - H(\mathbf{V}) - H(\mathbf{R}). \end{aligned}$$

Suppose that the message can be reconstructed by the extractor with probability 1. Then we have $H(\mathbf{M}|\mathbf{V},\mathbf{R}) = 0$. The inequality proven above imples that

$$H(\mathbf{M}) \leq H(\mathbf{V}) + H(\mathbf{R}). \tag{15}$$

Now suppose that the extractor is a black-box extractor. In this situation, we have that

$$H(\mathbf{R}) \leq \gamma \log_2 |\Delta|, \tag{16}$$

since there are γ possible challenges and each response is from the set Δ. The message m is a random vector in $(\mathbb{F}_q)^k$, so

$$H(\mathbf{M}) = k \log_2 q. \tag{17}$$

Therefore, combining (15), (16) and (17), we have the following result.

Theorem 6.1. *Suppose we have a keyed POR scheme where the message is a random vector in $(\mathbb{F}_q)^k$, there are γ possible challenges and each response is from the set Δ. Suppose that a black-box extractor succeeds with probability equal to 1. The the entropy of the verifier's storage, denoted $H(\mathbf{V})$, satisfies the inequality*

$$H(\mathbf{V}) \geq k \log_2 q - \gamma \log_2 |\Delta|.$$

We mentioned previously that Naor and Rothblum [11] proved a lower bound for a weaker form of POR-type protocol, termed an "authenticator". As noted in [6], the Naor-Rothblum bound also applies to POR schemes. Phrased in terms of entropy, their bound states that

$$H(\mathbf{M}) \leq H(\mathbf{V}) \times H(\mathbf{R}),$$

which is a weaker bound than (15).

In the case of an unkeyed scheme, the extractor is only given access to the proving algorithm. Therefore, $H(\mathbf{M}|\mathbf{R}) = 0$ if a black-box extractor succeeds with probability equal to 1. From this, it follows that $H(\mathbf{M}) \geq H(\mathbf{R})$ in this situation.

7 Conclusion

We have performed a comprehensive analysis of the extraction properties of unconditionally secure POR schemes, and established a methodology that is applicable to the analysis of further new schemes. What constitutes "good" parameters for such a scheme depends on the precise application, but our framework allows a flexible trade-off between parameters. One direction of possible future interest would be to consider the construction of further keyed POR schemes with a view to reducing the user's key storage requirements.

Acknowledgements

Thanks to Andris Abakuks and Simon Skene for some helpful discussions of statistics.

References

[1] G. Ateniese, R.C. Burns, R. Curtmola, J. Herring, L. Kissner, Z.N.J. Peterson and D.X. Song. Provable data possession at untrusted stores. *ACM Conference on Computer and Communications Security* (2007) 598–609.

[2] M. Blum, W.S. Evans, P. Gemmell, S. Kannan, and M. Naor. Checking the correctness of memories. *Algorithmica* **12** (1994), 225–244.

[3] K.D. Bowers, A. Juels and A. Oprea. Proofs of retrievability: theory and implementation. *Proceedings of the first ACM Cloud Computing Security Workshop* (2009), 43–54.

[4] G. Casella and R.L. Berger. *Statistical Inference*, Duxbury Press, 1990.

[5] F. Daly, D.J. Hand, M.C. Jones, A.D.Lunn, K.J. McConway. *Elements of Statistics*, Addison-Wesley, 1995.

[6] Y. Dodis, S. Vadhan and D. Wichs. Proofs of retrievability via hardness amplification. *Lecture Notes in Computer Science* **5444** (2009), 109–127 (TCC 2009).

[7] M. I. Husain, S. Ko, A. Rudra and S. Uurtamo. Storage enforcement with Kolmogorov complexity and list decoding. ArXiv report 1104/3025, 2011, `http://arxiv.org/abs/1104.3025`.

[8] A. Juels and B.S. Kaliski Jr. PORs: proofs of retrievability for large files. *ACM Conference on Computer and Communications Security* (2007), 584–597.

[9] M. Lillibridge, S. Elnikety, A. Birrell and M. Burrows. A cooperative internet backup scheme. *USENIX* (2003), 29–41.

[10] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*, North-Holland, 1977.

[11] M. Naor and G.N. Rothblum. The complexity of online memory checking. *Journal of the ACM* **56** (2009).

[12] H. Shacham and B. Waters. Compact proofs of retrievability. *Lecture Notes in Computer Science* **5350** (2008), 90–107 (ASIACRYPT 2008).

Appendix

Table 3: Notation used in this paper

q	order of underlying finite field
m	message
m_i	message block
$\mathcal{M}$	message space
k	length of a message
M	encoded message
$\mathcal{M}^*$	encoded message space
n	length of an encoded message
d	distance of the encoded message space
c	challenge
Γ	challenge space
γ	number of possible challenges
r	response
ρ	response function
r^M	response vector for encoded message M
Δ	response space
$\mathcal{R}^*$	response code
d^*	distance of the response code
$\mathcal{P}$	proving algorithm
$\mathsf{succ}(\mathcal{P})$	success probability of proving algorithm
$\widehat{m}$	message outputted by the Extractor
K	key (in a keyed scheme)
S	tag (in a keyed scheme)
dist	hamming distance between two vectors